

PAPER • OPEN ACCESS

Modelling non-markovian quantum processes with recurrent neural networks

To cite this article: Leonardo Banchi *et al* 2018 *New J. Phys.* **20** 123030

View the [article online](#) for updates and enhancements.



IOP | ebooksTM

Bringing you innovative digital publishing with leading voices to create your essential collection of books in STEM research.

Start exploring the **collection** - download the first chapter of every title for free.



OPEN ACCESS

RECEIVED

6 September 2018

REVISED

5 December 2018

ACCEPTED FOR PUBLICATION

10 December 2018

PUBLISHED

21 December 2018

Original content from this work may be used under the terms of the [Creative Commons Attribution 3.0 licence](#).

Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.



PAPER

Modelling non-markovian quantum processes with recurrent neural networks

Leonardo Banchi^{1,5}, Edward Grant^{2,4}, Andrea Rocchetto^{2,3}  and Simone Severini²¹ QOLS, Blackett Laboratory, Imperial College London, London SW7 2AZ, United Kingdom² Department of Computer Science, University College London, London WC1E 6EA, United Kingdom³ Department of Computer Science, University of Oxford, Oxford OX1 3QD, United Kingdom⁴ Rahko Limited, Kettering NN14 1UQ, United Kingdom⁵ Author to whom any correspondence should be addressed.E-mail: banchi.leonardo@gmail.com**Keywords:** Recurrent neural networks, Open quantum systems, Non-Markovian processes

Abstract

Quantum systems interacting with an unknown environment are notoriously difficult to model, especially in presence of non-Markovian and non-perturbative effects. Here we introduce a neural network based approach, which has the mathematical simplicity of the Gorini–Kossakowski–Sudarshan–Lindblad master equation, but is able to model non-Markovian effects in different regimes. This is achieved by using recurrent neural networks (RNNs) for defining Lindblad operators that can keep track of memory effects. Building upon this framework, we also introduce a neural network architecture that is able to reproduce the entire quantum evolution, given an initial state. As an application we study how to train these models for quantum process tomography, showing that RNNs are accurate over different times and regimes.

1. Introduction

Traditionally, in the physical sciences, the study of mathematical problems for which no analytic solution is available involves modelling methods leveraging a combination of approximation techniques (such as perturbation theory or semiclassical approaches) and the use of symmetries to reduce the complexity of the problem. Recently, advances in machine learning [1, 2], have caused a surge in popularity of data-driven approaches, which instead rely on computational techniques that exploit statistical correlations. Applications range from chaos theory [3] to high energy physics [4], eventually showing many applications and new perspectives in the quantum domain [5–8]. In particular, artificial neural networks (a class of learning methods inspired by the functioning of the brain) have been utilized in quantum many-body physics for ground state estimation [9], quantum state tomography [10, 11], classification of phases of matter [12], entanglement estimation [13], and to identify phase transitions [14]. Although the theoretical understanding of the effectiveness of these models is currently limited, some recent papers have established connections between neural networks and more standard frameworks such as renormalization group [15], tensor networks [16–18], and complexity theoretic tools [19]. Moreover, classical optimization techniques borrowed from supervised machine learning have been employed to optimize the dynamics of many-body systems [20–24] and parametric quantum circuits [25–31].

Open quantum systems [32, 33] present further challenges. Here, any modelling effort must take into account that the system interacts with a surrounding environment, whose microscopic details are usually unknown. The resulting effects can only be treated phenomenologically and significantly increase the complexity of the model, especially in the non-Markovian regime. In this regime, exact non-perturbative master equations [34] or quantum maps [35] operate by entangling the system with ancillary degrees of freedom whose dimension grows with time. This makes exact simulations extremely challenging even for low dimensional Hilbert spaces. Therefore, in larger systems, non-Markovian effects can only be modelled in an approximate fashion, e.g. by neglecting quantum correlations or by assuming weak couplings between system and environment. When these approximations are justified, phenomenological models with a reduced number of

parameters are normally accurate. However, these approximations are often violated, e.g. in quantum biology [34], so it is important to study alternative mathematical structures.

Machine learning methods offer new ideas for modelling non-Markovian effects when standard assumptions do not hold. The intuition behind these approaches is that, if there exists an efficient description of the system, this can be learnt from data without using explicit modelling assumptions. In particular, neural networks based learning techniques have shown that it is possible to learn complex functional dependencies in time series directly from the data, without trying to make theoretical assumptions that may be unjustified (such as the weak coupling limit) or hard to derive from phenomenological observations. In order for these methods to succeed, it is crucial to define the correct neural network structure that can quickly learn the underlying rule to reproduce the unknown functional form displayed in the data.

Here we investigate the ability of neural networks to model the non-Markovian evolution of open quantum systems, using a fixed number of parameters that does not grow with the number of time steps. We focus on Recurrent Neural Networks (RNNs), a type of artificial neural network specifically designed to model dynamical systems with possibly long range temporal correlations. In the quantum setting, RNNs have been previously employed for quantum control [36, 37]. We consider two main applications. In the first one, we define a master equation which has the mathematical simplicity of the Gorini–Kossakowski–Sudarshan–Lindblad (GKSL) master equation [38, 39], but is nonetheless able to model non-Markovian effects via the memory cells included in RNNs. In the second application, we define an RNN able to reproduce the entire quantum evolution given an initial state, without introducing any master equation. Our RNN-based frameworks share some similarity with collisional models [40–46] or ϵ -machines [47], where explicit memory effects are introduced by using ancillary quantum systems. However, there are notable differences. One of the advantages of RNNs is in their ability to learn directly from data how to compress complex time series with a few constant parameters. Moreover, there are different RNN architectures and it is possible to select the most appropriate one to model the expected long-range memory in the time evolution.

We will show that RNNs provide a convenient mathematical framework to define the so-called memory kernel [32], such that the resulting time-local master equation has a completely positive solution. The reconstruction of the memory kernel from quantum state sequences, namely from quantum process tomography, has been the subject of different studies in recent years [35, 48–52]. Most approaches considered in the literature start with some assumptions on the microscopic model, and then fits the free parameters given the experimental data. These assumptions are required to avoid having an exponentially growing number of free parameters for increasing number of time steps. However, these assumptions are normally uncontrollable, especially when the details of the microscopic model are unknown. Our main result is to show that RNN-based quantum processes and master equations are able to learn efficient representations of non-Markovian quantum evolutions directly from data sequences and without making any assumption on the underlying physical model. Indeed, even the Hamiltonian of the system can be learnt during this process. Note that, while the Hamiltonian can be learnt with alternative methods, e.g. Bayesian inference [53], the memory kernel is much more difficult, as its functional form, beyond the perturbative regime, is unknown. The main use of RNNs is therefore as a ‘compression’ method, to effectively model complex quantum correlations between system and environment with a constant and fixed number of parameters. As a relevant application for our technique, we consider quantum process tomography and show that RNNs are able to model the physical evolution of quantum systems over different regimes.

The paper is structured as follows. In section 2 we present relevant technical background. Specifically, in section 2.1 we introduce non-Markovian quantum processes and common master equations to describe them. In section 2.2 we introduce the RNN architecture employed in this paper. The main ideas are presented in section 3, where we introduce our RNN-based master equation (section 3.1) and RNN-based quantum process (section 3.2), with applications for process tomography (section 3.3). Numerical experiments are presented in section 4 and conclusions are drawn in section 5.

2. Background

2.1. Non-Markovian processes

Quantum systems, even the purest ones, are inevitably in contact with an environment. Because of this normally unknown interaction, quantum evolution deviates from the predictions of the Schrödinger equation, and different extensions have been proposed [32, 54]. When the interactions inside the environment happen on timescales much shorter than the internal timescales of the system, then a Markovian approximation is usually appropriate, and the evolution can be modelled via the GKSL master equation [38, 39]

$$\begin{aligned}\frac{\partial \rho(t)}{\partial t} &= \mathcal{L}[\rho(t)], \\ \mathcal{L}[\rho] &= -i[H, \rho] + \sum_{\mu} \left[L_{\mu} \rho L_{\mu}^{\dagger} - \frac{1}{2} \{ L_{\mu}^{\dagger} L_{\mu}, \rho \} \right],\end{aligned}\quad (1)$$

where H is the Hamiltonian, which models the noise-free case, and L_{μ} are called Lindblad operators. The superoperator \mathcal{L} is called Liouvillian. Mathematical properties of the above equation are well understood [33]. For any choice of H and L_{μ} the solution of the master equation $\mathcal{E}_t = e^{\mathcal{L}t}$ defines a completely positive trace preserving linear map, and is thus a mathematically well-posed mapping between states to states. With properly chosen Lindblad operators, the above master equation models the most general Markovian evolution [38, 39]. However, the Markovian approximation is not accurate in many situations, for instance when the interactions inside the environment have comparable strengths to the interactions inside the system [34, 55]. In that case the master equation has to be modified to take into account non-Markovian effects. One of the first and most accurate descriptions of non-Markovian evolution is the Nakajima–Zwanzig (NZ) master equation [32]

$$\frac{\partial \rho(t)}{\partial t} = -i[H, \rho(t)] + \int_0^t ds \mathcal{K}_{t-s}^{\text{NZ}}[\rho(s)] + \mathcal{I}(t), \quad (2)$$

where $\mathcal{K}_t^{\text{NZ}}$ is a superoperator, the so-called *memory kernel*, which describes the interaction with the environment, while $\mathcal{I}(t)$ is due to the initial correlations between system and environment. If system and environment are initially uncorrelated, then $\mathcal{I}(t) = 0$ for all t . It is clear that the above equation describes non-Markovian processes, because the state at time $t + dt$ depends not only on $\rho(t)$ but also on the states $\rho(s)$ for $s < t$. The NZ equation is at the basis of powerful Green function methods to study the spectral properties of the system [55–58], since the convolution disappears in the frequency domain. On the other hand, in the time domain the above equation is not easy to solve numerically. To avoid this problem, a different but equally accurate master equation has been proposed, the so-called time-convolutionless (TCL) master equation [32], which reads

$$\frac{\partial \rho(t)}{\partial t} = -i[H, \rho(t)] + \int_0^t ds \mathcal{K}_{t-s}^{\text{TCL}}[\rho(s)] + \mathcal{I}(t). \quad (3)$$

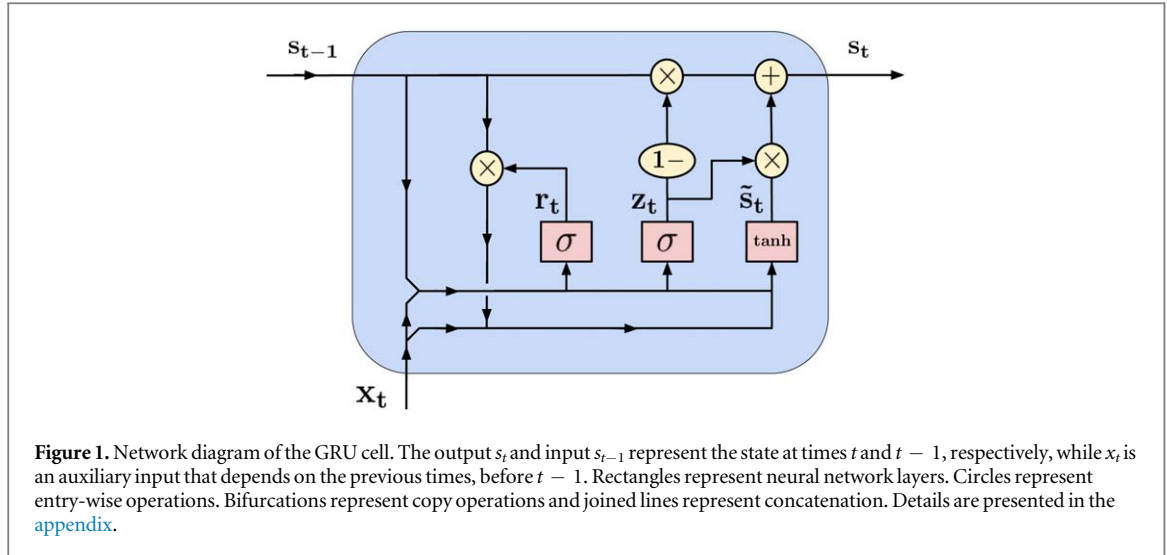
The main formal difference between equations (3) and (2) is that the whole history of states is fed into the NZ master equation, while in the TCL case the master equation explicitly depends on $\rho(t)$ only, and all non-Markovian effects are included into the memory kernel. As such, the non-Markovian nature of the process resulting from equation (3) is less obvious, but it is known that both NZ and TCL master equations can describe the same processes. Indeed, there are formal mappings between \mathcal{K}^{NZ} and \mathcal{K}^{TCL} such that both equations (3) and (2) produce the same physical evolution [32, 59].

Although TCL master equations are relatively easy to solve numerically, the main problem is that the interaction with the environment is normally unknown. In other terms, while H is typically well characterized experimentally, the memory kernel depends on environmental quantities such as the temperature, but also on the spectral properties of the environment which are normally unknown. Nonlinear spectroscopy can be used to find the spectral density [56], but the latter does not completely characterize the memory kernel, without introducing further assumptions. The most commonly employed approximation is the assumption of a weak coupling between system and environment, such that the memory kernel can be formally obtained using perturbation theory. However, there are cases where these approximations are not justified, e.g. in quantum biology [34], where the strength of the interaction with the environment is comparable with the internal interactions inside the system. To go beyond perturbation theory, we introduce then a RNN-based non-Markovian quantum process, where memory effects can be learnt directly from data.

2.2. Recurrent neural networks

RNNs are a class of neural networks designed to model data sequences like time series [60]. To understand their functioning, it is helpful to compare them with more standard feedforward networks. In feedforward neural networks the input data s^0 propagates throughout many intermediate (hidden) layers before reaching the final output layer. Here ‘propagate’ means that, step by step, the state $s^{\ell+1}$ of the $(\ell + 1)$ th layer is updated, given the state of ℓ th layer, as $s^{\ell+1} = f(W^{\ell}s^{\ell} + w^{\ell})$ where W^{ℓ} is a weight matrix, w^{ℓ} a weight vector and f is some nonlinear function. The state at the final layer of the network (output layer) depends on all the weight matrices and vectors. Training is performed by updating those weights such that the neural network learns some desired input–output relationship hidden in the data.

In case of temporal data, each input has also an explicit time dependence s_t^0 . Although, in principle, one could still use a giant feedforward network with these data, this is rarely the optimal choice, because the number of free parameters quickly increases with the number of time steps. RNNs solve this issue with a more advanced architecture which is tailored for temporal data. In RNNs, the update rule for the hidden layers at time t not only depends on the states s_t^{ℓ} , but also on the states at previous times. In other terms, the update rule is $s_t^{\ell+1} = f(W^{\ell}s_t^{\ell} + w^{\ell}, s_{t-1}^{\ell+1})$, where s_t^0 is the input temporal sequence. The free parameters W^{ℓ} and w^{ℓ} do not



depend on t , and memory of the past is taken into account by the function f , which compresses and saves relevant informations of previous sequences into memory cells. This architecture allows RNNs to learn temporal sequences using a relatively small number of parameters, even when the temporal data has long-range memory effects. The mapping between s_t^ℓ and $s_t^{\ell+1}$ defines a RNN cell.

In this work we use a variant of RNN called Gated Recurrent Unit (GRU), whose basic cell is shown in figure 1 and discussed in the [appendix](#). GRUs use a gating mechanism that allows them to better model long-term dependencies than more simple RNNs [61]. GRUs are based on a type of RNN cell called Long Short-Term Memory (LSTM), but can be more efficient than LSTMs for comparable performance [62, 63]. GRU and LSTM are commonly used and achieve state of the art performance for sequence modelling across multiple domains, including machine translation, image captioning and forecasting [64].

The GRU state s_t is a linear interpolation of the previous state s_{t-1} and a candidate state \tilde{s}_t , which depends on the auxiliary input x_t . The input x_t^j for a depth j cell at time t is the state from the cell in the previous layer s_t^{j-1} . GRU cells can be stacked to form a deep GRU network. More details can be found in the [appendix](#).

3. Main idea

In this section we present the three main contributions of this paper which all leverage on the modelling capabilities of RNNs to describe non-Markovian dynamics of open quantum systems. First, we describe a master equation approach. Second, we use RNNs to predict the time evolution of quantum state under a non-Markovian quantum process. Third, we show how these two techniques can be utilized to perform quantum process tomography.

3.1. RNN quantum master equation

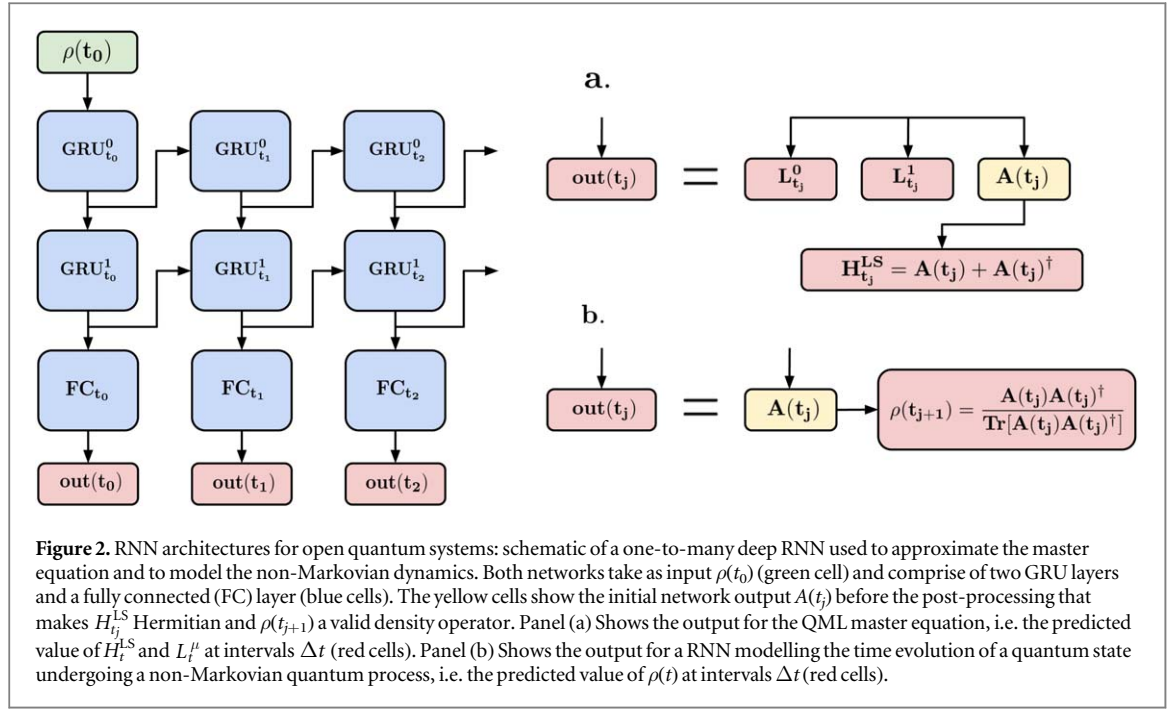
We postulate a quantum evolution similar to equation (1), namely

$$\frac{\partial \rho(t)}{\partial t} = \mathcal{L}_{\leq t}[\rho(t)], \quad (4)$$

where the notation $\leq t$ refers to a superoperator that not only depends on t , but also on the entire history before time t , as in the TCL master equation (3). A convenient choice is then that of the GKSL form

$$\begin{aligned} \mathcal{L}_{\leq t}[\rho] = & -i[H + H_{\leq t}^{\text{LS}}, \rho] \\ & + \sum_{\mu} \left[L_{\leq t}^{\mu} \rho L_{\leq t}^{\mu\dagger} - \frac{1}{2} \{L_{\leq t}^{\mu\dagger} L_{\leq t}^{\mu}, \rho\} \right], \end{aligned} \quad (5)$$

where H^{LS} is a ‘Lamb-shift’ term, namely a correction to the Hamiltonian induced by the environment, while L^{μ} are Lindblad operators. The reason for this choice is that for small enough Δ_t the time evolution is simply given by $\rho(t + \Delta_t) \approx e^{\Delta_t \mathcal{L}_{\leq t}}[\rho(t)]$, and since $\mathcal{L}_{\leq t}$ is in the GKSL form, $e^{\Delta_t \mathcal{L}_{\leq t}}$ is a completely positive trace preserving quantum channel, mapping states to states. If H^{LS} and L^{μ} are simply time-dependent functions, namely they only depend on t and not on previous times, then the dynamics generated by above master equation is always Markovian [59]. The main idea of this work is to use a RNN to define each Lindblad operator $L_{\leq t}^{\mu}$ and the correction Hamiltonian $H_{\leq t}^{\text{LS}}$, see figure 2(a). In order to ensure the Hermiticity of the $H_{\leq t}^{\text{LS}}$ operator, we construct it as



$H_{t_j}^{LS} = A(t_j) + A(t_j)^\dagger$, where $A(t_j)$ is the output of the network and $A(t_j)^\dagger$ its conjugate transpose. Since in RNNs the predicted output at time t depends on the entire history at previous times, this parametrization is expected to accurately reproduce genuinely non-Markovian effects, even with possible long-range dynamical correlations. The master equation then resembles a TCL master equation (3), but where the complicated memory superoperator is expressed via a simpler GKSL form with RNNs. We call our equation (5) the Quantum Recurrent Neural (QRN) Master equation. Similarly, we call the operators $L_{\leq t}^\mu$ recurrent Lindblad operators. A schematics of the resulting neural network is shown in figure 2(a).

3.2. RNN quantum processes

For any initial time t_0 the mapping between the initial state $\rho(t_0)$ and the state at time t , namely

$$\rho(t) = \mathcal{E}(t, t_0)[\rho(t_0)], \quad (6)$$

defines a completely positive map, assuming no initial correlation with the environment. For any intermediate times $t_0 < \tau < t$ the mapping $\mathcal{E}(\tau, t_0)$ is always completely positive. When also $\mathcal{E}'(t, \tau) := \mathcal{E}(t, t_0)\mathcal{E}(\tau, t_0)^{-1}$ is completely positive, then the mapping is called *divisible* [65]. Divisibility is another way of characterising Markovianity, as quantum processes obtained from the GKSL master equation are always divisible with $\mathcal{E}(t', t) = \mathcal{T} \exp \left(\int_t^{t'} \mathcal{L}_s ds \right)$.

In the previous section we have introduced the QRN master equation and shown that it can model non-Markovian effects, even when the evolution between intermediate steps is completely positive. This is possible because the RNN keeps a compressed record of the previous evolution. Using the formalism of the previous section we can indeed write

$$\mathcal{E}(t, t_0) = \overleftarrow{\prod}_j \mathcal{E}^{\text{QRN}}(t_j + \Delta t, t_j) + \mathcal{O}(\Delta t^2), \quad (7)$$

where $\overleftarrow{\prod}_j X_j$ is the ordered product $\cdots X_3 X_2 X_1$. Moreover,

$$\mathcal{E}^{\text{QRN}}(t_j + \Delta t, t_j) = e^{\Delta t \mathcal{L}_{\leq t_j}} \quad (8)$$

is completely positive, being the operator exponential of a Liouvillian, but depends on the previous evolution, via the recurrent Lindblad operators. As such, the total map (7) is not divisible.

Based on this analogy, we can now drop the master equation and define a non-Markovian process as

$$\mathcal{E}(t, t_0) = \overleftarrow{\prod}_j \mathcal{E}_{\leq t_j}^{\text{RNN}}(t_{j+1}, t_j), \quad (9)$$

where $\mathcal{E}_{\leq t_j}^{\text{RNN}}(t_{j+1}, t_j)$ is completely positive, but depends on the entire history before t_j . Each $\mathcal{E}_{\leq t_j}^{\text{RNN}}(t_{j+1}, t_j)$, being completely positive, outputs a valid quantum state at intermediate times $\rho(t_j)$ and, being a RNN, then updates its

internal memory. Complete positivity can be ensured by using the Kraus decomposition or the environmental representation.

For better comparison with the QRN master equation, in this work we use the simpler strategy shown in figure 2(b), where we use the output $A(t_j)$ of the network to define a density operator via $\rho(t_{j+1}) = A(t_j)A(t_j)^\dagger / \text{Tr}[A(t_j)A(t_j)^\dagger]$. This ensures that the states $\rho(t_j)$ are valid density operators, throughout the entire evolution.

3.3. Application: quantum process tomography

In this work we apply our QRN master equations and RNN quantum processes for quantum process tomography. We consider the following setup. We assume that the quantum system can be initialized in different initial states $\rho_\alpha(0)$ where $\alpha = 1, \dots, N_I$, for some number N_I . For each initialization, we assume that it is possible to perform full-quantum tomography at some time steps $0 \leq t_j \leq T$ for $j = 1, \dots, N_T$ and some N_T . After this procedure we are able to reconstruct the time evolutions $\rho_\alpha(t_j)$ for different times and initializations. Here we consider uniformly separated times where $t_j = j \Delta_T$ and $\Delta_T = T/N_T$, though it is straightforward to generalize this procedure to non-uniform sequences. Each state tomography requires $\mathcal{O}(d^2)$ measurements, where d is the dimension of the system's Hilbert space, so the total cost of reconstructing these sequences is $\mathcal{O}(d^2 N_T N_I)$. Once these sequences are obtained, we use them to train the neural network.

We consider two cases. In the first one we train a RNN to learn the quantum state evolution, as shown in figure 2(b). Here we assume no knowledge about the system's Hamiltonian or interaction with the environment. Training is then performed by minimising a cost function

$$J_p = \frac{1}{N_I N_T} \sum_{\alpha=1}^{N_I} \sum_{j=1}^{N_T} \|\rho_\alpha(t_j) - \tilde{\rho}_\alpha(t_j)\|, \quad (10)$$

where $\rho_\alpha(t_j)$ are the training data, $\tilde{\rho}_\alpha(t_j)$ are the states outputted by the RNN, and $\|\cdot\|$ is any operator norm.

In the second application we aim at reconstructing $H_{\leq t}^{LS}$ and the recurrent Liouvillian operators $L_{\leq t}^\mu$ entering in the QRN master equation, where, on the other hand, we assume that the Hamiltonian H is known. The latter assumption can always be relaxed, as Hamiltonian evolution can be fully included into the correction Hamiltonian H^{LS} , which is learnt from the data. In the QRN master equation, the operators $H_{\leq t}^{LS}$ and $L_{\leq t}^\mu$ are obtained from the output of the RNN, as shown in figure 2(b). To train the RNNs to predict $H_{\leq t}^{LS}$ and $L_{\leq t}^\mu$ given a starting state $\rho_\alpha(t=0)$ we propose the use of the differential equation (4) to define a cost function. For classical time series, a similar approach [66] has been employed for extracting natural laws from experimental data.

To explain our idea, let us first consider the opposite scenario, where the differential equation (4), with all of its operators, is already known, while the states $\rho(t_j)$ are not. In this common case, the states $\rho(t_j)$ at different times are evaluated from the numerical integration of the master equation. The latter can be obtained with an n -order Runge–Kutta integrator [67] which, in general, can be formally written as

$$\rho_\alpha(t_{j+1}) = \rho_\alpha(t_j) + \Delta_T \mathcal{L}_{\leq t_j}^{RK,n}[\rho_\alpha(t_j)], \quad (11)$$

where $\mathcal{L}_{\leq t}^{RK,n}$ is n th order Runge–Kutta integration step, which can be explicitly obtained for any n (see e.g. [67]). For instance, to the first order $\mathcal{L}_{\leq t}^{RK,1}$ is simply $\mathcal{L}_{\leq t}$. To summarize, when $H_{\leq t}^{LS}$ and $L_{\leq t}^\mu$ are known, we can use a Runge–Kutta integrator to obtain the time evolution $\rho(t_j)$.

We now consider the opposite problem, namely where many time sequences $\rho_\alpha(t_j)$ are already known, while the operators $H_{\leq t}^{LS}$ and $L_{\leq t}^\mu$ in the QRN master equation are not. This is like assuming that the solutions of a differential equation are known and from them we want to reconstruct the differential equation itself. Based on the analogy with numerical integration via Runge–Kutta, we propose to use the following cost function:

$$J = \frac{1}{N_I N_T} \sum_{\alpha=1}^{N_I} \sum_{j=1}^{N_T} \|\rho_\alpha(t_{j+1}) - \rho_\alpha(t_j) - \Delta_T \mathcal{L}_{\leq t_j}^{RK,n}[\rho_\alpha(t_j)]\|_F^2, \quad (12)$$

where $\|\cdot\|_F$ is the Frobenius norm. The intuitive idea behind the above cost function is that of making the measured data sequences as close as possible to those coming from the numerical solution of a master equation. Minimizing the cost function is then equivalent to finding the best QRN master equation compatible with the measured sequences $\rho_\alpha(t_j)$. It is expected that a higher order integrator (large n) performs better, especially for larger Δ_T , but requires heavier numerical computations.

4. Numerical experiments

4.1. Learning quantum state sequences

We mimic experimental data by numerically generating sequences of states, which are then used to train the neural network. To generate the training data, we consider a simple yet important model of spontaneous decay of a two-level system [54, 68], described by the master equation

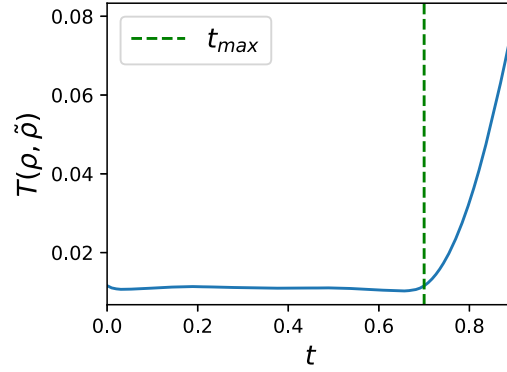


Figure 3. (EXP1) Learning the evolution of a noisy two level system: the mean trace distance $T(\rho(t), \tilde{\rho}(t)) = \frac{1}{N_p} \sum_{\beta} D(\rho_{\beta}(t), \tilde{\rho}_{\beta}(t))$ between true states $\rho_{\beta}(t_j)$ and predicted states $\tilde{\rho}_{\beta}(t_j)$ as a function of time. The time t_{\max} is the maximum time used during training. All the experiments run with the following parameters $\omega = 1$ for equation (13) and $\lambda = 2, \gamma_0 = 0.5$ for equation (14), and a discretization interval of $\Delta t = 0.01$. The simulations run over 3000 training examples and predictions were tested over 1000 test examples. All the evolutions run for a time of $t_{\max} = 0.7$.

$$\begin{aligned} \frac{\partial}{\partial t} \rho(t) = & -i[\omega \sigma_z, \rho(t)] \\ & + \gamma(t) \left(\sigma^- \rho(t) \sigma^+ - \frac{1}{2} \{ \sigma^+ \sigma^-, \rho(t) \} \right), \end{aligned} \quad (13)$$

where $\{x, y\} = xy + yx$, σ^{α} for $\alpha = x, y, z$ are the Pauli matrices, $\sigma^{\pm} = (\sigma^x \pm i\sigma^y)/2$, ω is the Rabi frequency of oscillations around the z axis. The parameter

$$\gamma(t) = \frac{2\gamma_0 \lambda \sinh(\eta t/2)}{\eta \cosh(\eta t/2) + \lambda \sinh(\eta t/2)}. \quad (14)$$

is the decay rate with $\eta = \sqrt{\lambda^2 - 2\gamma_0 \lambda}$. When $\lambda > 2\gamma_0$ the function $\gamma(t)$ is always positive, so equation (13) takes the GKSL form (1) and, as such, defines a Markovian evolution. On the other hand, for $\lambda < 2\gamma_0$ the function $\gamma(t)$ can be negative and the dynamics displays non-Markovian effects [54, 69, 70]. The above two level system can also model an excitation energy transfer in biological dimers [71].

We use the above model to obtain training data for the RNN architecture shown in figure 2(b). Each training sequence $\rho_{\alpha}(t_j)$, for $\alpha = 1, \dots, N_t$, has been obtained by first choosing a random initial state $\rho_{\alpha}(0)$ and then solving the master equation (13) to get the states $\rho_{\alpha}(t_j)$ at subsequent times, up to a maximal time t_{\max} . These data sequences were then used to train a GRU neural network, by minimising the cost function (10). After training, we test the accuracy of the neural network by generating a new sequence of states $\rho_{\beta}(t_j)$, and the RNN prediction $\tilde{\rho}_{\beta}(t_j)$, for $\beta = 1, \dots, N_p$. As before, $\rho_{\beta}(t_j)$ is obtained by selecting a random initial state $\rho_{\beta}(0)$ and solving equation (13), possibly for longer times than t_{\max} . On the other hand, the predicted evolution $\tilde{\rho}_{\beta}(t_j)$ is obtained by feeding the initial state $\rho_{\beta}(0)$ to the RNN to get the entire temporal sequence. The two evolutions are compared with the trace distance $D(\rho, \tilde{\rho}) = \frac{1}{2} \text{Tr}|\rho - \tilde{\rho}|$. In particular, we study the average trace distance $T(\rho(t), \tilde{\rho}(t)) = \frac{1}{N_p} \sum_{\beta} D(\rho_{\beta}(t), \tilde{\rho}_{\beta}(t))$ as a function of time.

In figure 3 we show a numerical solution of this numerical experiment (EXP1), where we can see that, once trained, the RNN is able to predict the evolution of a known starting state, up to the maximal training time t_{\max} . On the other hand, and as expected, the accuracy of the RNN prediction rapidly deteriorates for $t > t_{\max}$.

4.2. Learning the master equation: a simple case

In figure 4 we show the solution of a second numerical experiment (EXP2) obtained with the same training set of the first experiment (EXP1), discussed in figure 3. While EXP1 uses a RNN to model the entire quantum evolution, EXP2 uses the RNN to define a QRN master equation, following section 3.3. Training is then performed by minimising the QRN cost function (12), where for simplicity we assume a first-order Runge–Kutta integrator ($n = 1$) and a single recurrent Lindblad operator ($\mu \equiv 1$). In EXP2 we have chosen a Markovian regime so that the entire evolution can be modelled via a GKSL master equation (1). In this regime, the RNN Lindblad operators can be approximated as a simple time-dependent function, so the minimization of the cost function (12) is equivalent to learning standard Lindblad operators. In figure 4 we compare the predicted recurrent Lindblad operators L_{μ} as a function of time, with respect to the real ones defined in equation (13). In

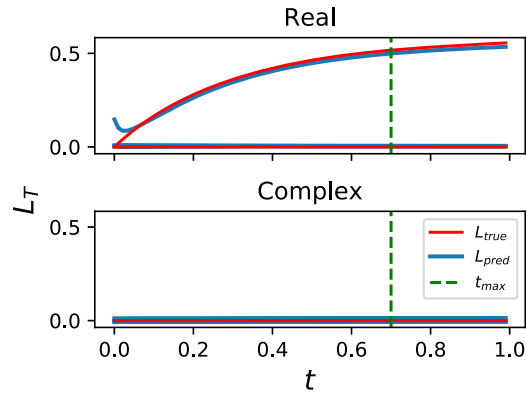


Figure 4. (EXP2) Learning the Lindblad operator describing the Markovian evolution of a two level system: the learned and known entries of $\frac{1}{\alpha} \sum_{\alpha} L^{\mu} \equiv 1$ are compared for the Markovian evolution of a two-level system. We plot the time evolution of every entry of the matrix. Real and complex entries are plotted separately. All the experiments run with the following parameters $\omega = 1$ for equation (13) and $\lambda = 2, \gamma_0 = 0.5$ for equation (14), and a discretization interval of $\Delta t = 0.01$. The simulations run over 1500 training examples and predictions were tested over 2500 test examples. All the evolutions run for a time of $t_{\max} = 0.7$. The Frobenius norm squared was used as distance between the matrices.

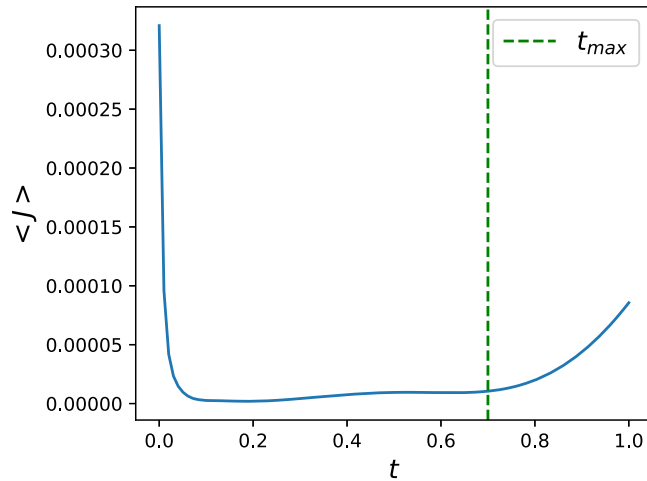


Figure 5. (EXP2) Average cost function value (12) for different times. The parameters are the same of figure (4).

the system under study all entries of the Lindblad operators are zero (hence the flat lines in the figure) apart from one value. As we can see, the prediction is remarkably accurate at all times, even beyond the training time $t > t_{\max}$. The error in the prediction is shown in figure 5, by plotting the cost function (12) for different times. EXP2 shows that the RNN was able to learn the evolution of the Lindblad operator L_t for $t = 0.1$ to $t_{\max} = 0.7$. In addition, the RNN was able to predict L_t for $t > t_{\max}$ which was the last time step used during training.

4.3. Learning the master equation: non-Markovian case

In this section we focus on the non-Markovian regime, where there are non-trivial memory effects that the RNN has to learn and reproduce. As in section 4.1, we consider state sequences generated numerically by solving a master equation. However, unlike our previous treatment, here we consider a more complicated non-Markovian model of the environment, which includes back-scattering effects. Back-scattering can refer, for instance, to a photon emitted to the environment that comes back at later times. As such, the information transferred to the environment is not completely lost. To model these non-Markovian effects we consider two qubits evolving with the following master equation

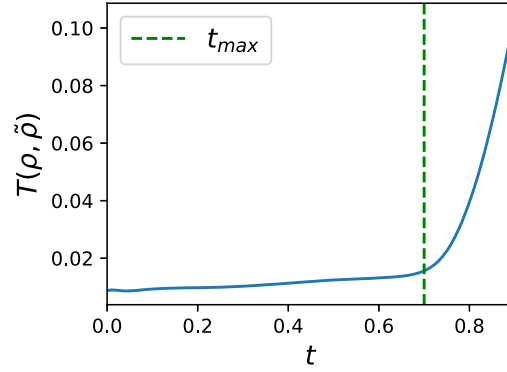


Figure 6. (EXP3) Learning the evolution of a noisy two level non-Markovian system: the mean trace distance $T(\rho(t), \tilde{\rho}(t)) = \frac{1}{N_p} \sum_{\beta} D(\rho_{\beta}(t), \tilde{\rho}_{\beta}(t))$ between true states $\rho_{\beta}(t_j)$ and prediction states $\tilde{\rho}_{\beta}(t_j)$ as a function of time. The time t_{\max} is the maximum time used during training. The experiments run with the following parameters $\lambda^{(1)} = 2$, $\gamma_0^{(1)} = 0.5$ and $\lambda^{(2)} = 1$, $\gamma_0^{(2)} = 0.2$ for equation (15), $\omega = 1$ for equation (16), and a discretization interval of $\Delta t = 0.01$. The simulations run over 3000 training examples and predictions were tested over 1000 test examples. All the evolutions run for a time of $t_{\max} = 0.7$.

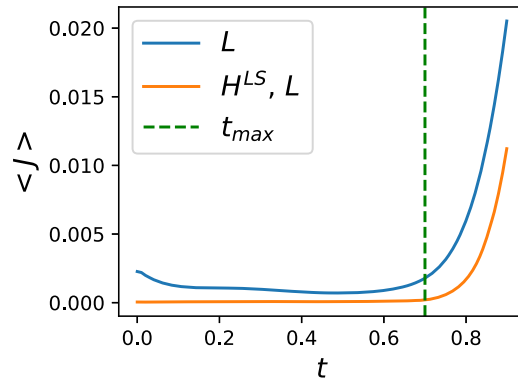


Figure 7. (EXP4) Learning a non-Markovian memory kernel: average value of the cost function $\langle J \rangle$ on unseen examples as a function of time for a learned memory kernel with only the Lindblad Operator $L_{\leq t}^{\mu}$ and with both the Lindblad Operator and the Lamb-shift Hamiltonian $H_{\leq t}^{LS}$. The parameters are the same of figure 6. The Frobenius norm squared was used as distance between the matrices.

$$\begin{aligned} \frac{\partial}{\partial t} \rho_{12}(t) = & -i[H_{12}, \rho_{12}(t)] \\ & + \sum_{i=1}^2 \gamma_i(t) \left(\sigma_i^- \rho_{12}(t) \sigma_i^+ - \frac{1}{2} \{ \sigma_i^+ \sigma_i^-, \rho_{12}(t) \} \right), \end{aligned} \quad (15)$$

where $\gamma_i(t)$ has the same functional form of equation (14) (but we add a superscript (i) to the parameters $\gamma_0^{(i)}$ and $\lambda^{(i)}$ that refers to qubit index) and the two-qubit Hamiltonian is

$$H_{12} = \omega \sigma_z \otimes I + c_1 \sigma_x \otimes \sigma_x + c_2 \sigma_y \otimes \sigma_y + c_3 \sigma_z \otimes \sigma_z, \quad (16)$$

where $c_1 = 0.3242$, $c_2 = 0.6723$, and $c_3 = 0.1353$. We numerically solve equation (15) to get the data sequence $\rho_{12}^{\alpha}(t_j)$, where $\alpha = 1, \dots, N_l$ indexes the different solutions obtained with different initial states. From these two qubit solutions we define then the state sequences as $\rho_{\alpha}(t_j) = \text{Tr}_2[\rho_{12}^{\alpha}(t)]$. In other terms, qubit 1 is the principal system, while qubit 2 is an ancillary system. Because of the coherent interaction H_{12} between qubit 1 and qubit 2, with this approach we can mimic the back-action of the environment onto the system.

We divide our numerical results into different experiments. In EXP3, shown in figure 6, we train a RNN to fully reproduce the entire quantum evolution, following the discussion of section 3.2. We see that in spite of non-Markovian effects, the resulting error is comparable to that of the Markovian case (EXP1) shown in figure 3.

In figure 7 we use the same training data of EXP3 to run a new experiment (EXP4) where we train a QRN master equation, by minimising the cost function (12). We consider two cases: in the first one the RNN outputs a single recurrent Lindblad operator $L^{\mu=1}$. In the second one, the RNN outputs both model both a recurrent Lindblad operator $L^{\mu=1}$, and the renormalized Hamiltonian, namely the Lamb-shift term H^{LS} . We note that, with a single recurrent Lindblad operator, the error is slightly larger than that of the Markovian case, shown in figure 4. However, the resulting error is very low when we include also the correction Hamiltonian H^{LS} .

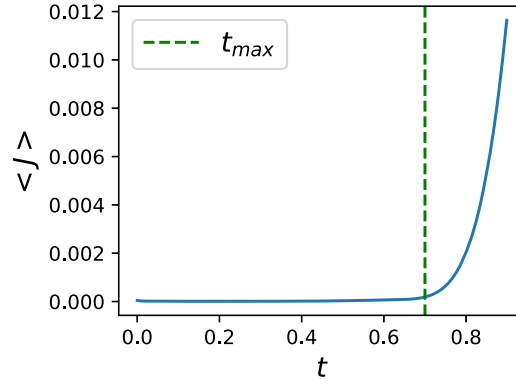


Figure 8. (EXP5) Learning a non-Markovian master equation with different ω values: average value of the cost function $\langle J \rangle$ on unseen examples as a function of time for a learned memory kernel with both the Lindblad operator $L_{\leq t}^{\mu}$, where $\mu = 2$ and the Lamb-shift Hamiltonian $H_{\leq t}^{LS}$. t_{max} specifies the maximum time in the range of times used for training. All the experiments run with the following parameters $\lambda^{(1)} = 2$, $\gamma_0^{(1)} = 0.5$ and $\lambda^{(2)} = 1$, $\gamma_0^{(2)} = 0.2$ for equation (15), ω uniformly sampled in the interval $[0.5, 1.5]$ for equation (16), and a discretization interval of $\Delta t = 0.01$. The simulations run over 3000 training examples and predictions were tested over 1000 test examples. All the evolutions run for a time of $t_{max} = 0.7$. The Frobenius norm squared was used as distance between the matrices.

Comparing figure 7 with the Markovian case, figure 4, one can see that the error grows faster in the non-Markovian regime after the training time t_{max} . Overall, in figure 7 we see that the RNN which learned both the recurrent Lindblad operator and the Lamb-shift Hamiltonian performed best, and was better able to predict the memory kernel at times greater than those seen during training.

Finally, in figure 8 we run a different numerical experiment (EXP5). In EXP5 the training set is composed by data sequences where the qubit frequency ω in equation (16) is not fixed, but rather uniformly sampled from 0.5 to 1.5. The sampled frequency is used as an extra input to the RNN. This corresponds to the experimentally relevant case where the qubit frequency can be externally tuned to a known value. Uncertainty about this frequency can be estimated from the correction Hamiltonian H^{LS} , which is learned from the data. In figure 8 we see that the error in EXP5 is remarkably low. Based on the success of this numerical experiment, we propose the following general strategy to introduce prior knowledge about the system. We can define a RNN where the known properties of the system, e.g. its Hamiltonian, are added as an extra input. Training is performed using datasets of quantum state sequences and their respective Hamiltonian, where the latter is sampled from the space of experimentally relevant Hamiltonians. The remarkable accuracy shown figure 8 suggests that this procedure forces the RNNs to better explore the manifold of quantum state sequences, to produce a more accurate and robust prediction.

5. Conclusions and perspectives

We have studied quantum state evolution using RNNs. We have shown that even when the system is interacting with a complicated surrounding environment, RNNs offer an accurate and robust tool for modelling and reconstructing the quantum evolution, both in the Markovian and non-Markovian regime, and even when there are back-scattering effects from the environment.

We have introduced two approaches for modelling open quantum systems with RNNs. In the first one, a deep RNN is trained to learn the entire quantum evolution, namely to reproduce the time sequence $\rho(t_i)$ given an initial state $\rho(0)$. In the second approach, we use a deep RNN to define a non-Markovian master equation where the memory kernel takes a convenient mathematical form, namely that of the GKSL equation. In our master equation the non-Markovian memory effects are taken into account by the structure of the RNN cells. The observed success of our approaches stems from the ability of RNNs to learn temporal sequences, where the future depends on the entire past. Our RNN-based master equation can be used as a convenient mathematical framework to write non-Markovian memory kernels without resorting to perturbative treatments, and to learn their unknown parameters directly from the data.

Many extensions of our work are possible. Many-body systems with exponentially large Hilbert spaces could be considered by using RNNs which output a compressed representation of the state, such as tensor networks [72, 73], restricted Boltzmann machines [10] or variational autoencoders [19]. The predictions after the training time can be improved by considering higher order Runge–Kutta integrators and by modifying the structure of the RNN cell to have a fixed point in the infinite time limit, as expected from physical principles. Although RNNs

proved to be a powerful tool for modelling open quantum systems, the machine learning literature offers a number of other methods, such as Kalman Filters or models with Gaussian Process transitions, that appear particularly promising. For example, Gaussian Process State Space Models [74] allow one to model prior information on the system and return Bayesian estimates of the uncertainties. Both these features are desirable in a quantum context: prior knowledge on the system, such as the form of the noise-free Hamiltonian, may be used to further reduce the complexity of the model, while approximate values for the uncertainties might enable better control of experimental inaccuracies. It would be interesting to define quantum master equations based on these tools and to check their performances against our QRN master equation.

Finally, in more physical terms, it would be interesting to study what happens when *partial* information about the system is available. An experimentally relevant case is when the initial state $\rho(0)$ is known, but one has access to a limited set of expectation values $\langle A_k \rangle_{\rho(t_j)}$, where the observables A_k are not enough to tomographically reconstruct the states $\rho(t_j)$. This possibility may be considered, using our framework, by introducing a cost function between expectation values, rather than between density operators. A further challenge is then to model the disturbance of the measurement onto the system, namely the wave function collapse. This can be done using the process tensor formalism [35], which provides an avenue for generalising our approach in the presence of feedback. It would be interesting to study the performance of RNNs to model these experimentally relevant quantum evolutions.

Acknowledgments

The authors would like to thank G Carleo, A D Ialongo, and M Paternostro for valuable discussions. LB is supported by the UK EPSRC grant EP/K034480/1. EG is supported by the UK EPSRC grant EP/P510270/1. AR is supported by an EPSRC DTP Scholarship and by QinetiQ. SS is supported by the Royal Society, EPSRC, the National Natural Science Foundation of China, and the grant ARO-MURI W911NF17-1-0304 (US DOD, UK MOD and UK EPSRC under the Multidisciplinary University Research Initiative). We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. Part of this work was conducted while AR and SS were at the Institut Henri Poincaré in Paris. EG would further like to acknowledge support from Rahko Limited.

Appendix. Gated recurrent unit

The GRU state s_t is a linear interpolation of the previous state s_{t-1} and a candidate state \tilde{s}_t . The candidate state is a function of the cell input x_t and the previous state given by:

$$\begin{aligned} z_t &= \sigma(W_z x_t + U_z s_{t-1} + b_z) , \\ r_t &= \sigma(W_r x_t + U_r s_{t-1} + b_r) , \\ \tilde{s}_t &= \tanh(W x_t + U(r_t \circ s_{t-1}) + b) , \\ s_t &= (1 - z_t) \circ s_{t-1} + z_t \circ \tilde{s}_t , \end{aligned} \tag{A1}$$

where \circ denotes the entry-wise (Hadamard) product, σ denotes the sigmoid function, x_t is the input to the cell and W , U and b are trainable parameters, which do not explicitly depend on t . GRU cells can be stacked to form a deep GRU network. The input x_t^j for a depth j cell at time t is the state from the cell in the previous layer s_t^{j-1} .

The GRU state s_t can be thought of as a mixture of the previous state s_{t-1} and the candidate state \tilde{s}_t . The weighting of the mixture is controlled by z_t which is known as the *update gate*. We can see that if the entries of z_t are zero then the candidate state is ignored and the previous state becomes the new state.

The candidate state \tilde{s}_t is a function of the previous state s_{t-1} and the current input x_t . The relative contribution of the previous state to the candidate state is controlled by r_t which is known as the *reset gate*.

A more detailed discussion of GRU cells can be found in [61].

Note that the output of the network is a real vector. In order to encode complex matrices into the RNN we use the following procedure. Assume that we want to encode a matrix $M \in \mathbb{C}^{m \times m}$. M can be decomposed into a real and imaginary part $M = M_{\text{Re}} + iM_{\text{Im}}$. We require the output of the RNN to be a vector $o \in \mathbb{R}^{2m^2}$ such that the first m^2 elements encode in row-major order the entries of M_{Re} and the second m^2 elements encode in row-major order the entries of M_{Im} .

All GRU networks used in this work consisted of 2 GRU layers with output dimension = 40, followed by 1 fully connected layer. The Adam optimizer was used for training with initial learning rate 0.01 [75] and batches of 32 examples. Training was stopped after each example was seen 60 times. In EXP2 the fully connected weights were regularized using weight decay [76], with the L2 penalty factor set to 0.001. The networks were implemented using the Keras 2.2.0 and Tensorflow 1.8 frameworks [77, 78]. Network weights were initialized using the Keras defaults.

ORCID iDs

Andrea Rocchetto  <https://orcid.org/0000-0002-5216-0095>

References

- [1] Bishop C M 2006 *Pattern Recognition and Machine Learning (Information Science and Statistics)* (Berlin: Springer)
- [2] LeCun Y, Bengio Y and Hinton G 2015 *Nature* **521** 436
- [3] Pathak J, Hunt B, Girvan M, Lu Z and Ott E 2018 *Phys. Rev. Lett.* **120** 024102
- [4] Komiske P T, Metodiev E M and Schwartz M D 2017 *J. High Energy Phys.* **JHEP01(2017)110**
- [5] Schulm M, Sinayskiy I and Petruccione F 2015 *Contemp. Phys.* **56** 172
- [6] Biamonte J, Wittek P, Pancotti N, Rebentrost P, Wiebe N and Lloyd S 2017 *Nature* **549** 195
- [7] Ciliberto C, Herbster M, Ialongo A D, Pontil M, Rocchetto A, Severini S and Wossnig L 2018 *Proc. R. Soc. A* **474** 20170551
- [8] Dunjko V and Briegel H J 2018 *Rep. Prog. Phys.* **81** 074001
- [9] Carleo G and Troyer M 2017 *Science* **355** 602
- [10] Torlai G, Mazzola G, Carrasquilla J, Troyer M, Melko R and Carleo G 2018 *Nat. Phys.* **14** 447
- [11] Yu S *et al* 2018 arXiv:1808.09241
- [12] van Nieuwenburg E P, Liu Y-H and Huber S D 2017 *Nat. Phys.* **13** 435
- [13] Gray J, Banchi L, Bayat A and Bose S 2018 *Phys. Rev. Lett.* **121** 150503
- [14] Carrasquilla J and Melko R G 2017 *Nat. Phys.* **13** 431
- [15] Mehta P and Schwab D J 2014 arXiv:1410.3831
- [16] Cohen N, Sharir O, Levine Y, Tamari R, Yakira D and Shashua A 2017 arXiv:1705.02302
- [17] Levine Y, Sharir O, Cohen N and Shashua A 2018 arXiv:1803.09780
- [18] Chen J, Cheng S, Xie H, Wang L and Xiang T 2018 *Phys. Rev. B* **97** 085104
- [19] Rocchetto A, Grant E, Strelchuk S, Carleo G and Severini S 2018 *npj Quantum Inf.* **4** 28
- [20] Banchi L, Pancotti N and Bose S 2016 *npj Quantum Inf.* **2** 16019
- [21] Innocenti L, Banchi L, Ferraro A, Bose S and Paternostro M 2018 arXiv:1803.07119
- [22] Eloie L, Banchi L and Bose S 2018 *Phys. Rev. A* **97** 062321
- [23] Las Heras U, Alvarez-Rodriguez U, Solano E and Sanz M 2016 *Phys. Rev. Lett.* **116** 230504
- [24] Zahedinejad E, Ghosh J and Sanders B C 2016 *Phys. Rev. Appl.* **6** 054005
- [25] Verdon G, Pye J and Broughton M 2018 arXiv:1806.09729
- [26] Grant E, Benedetti M, Cao S, Hallam A, Lockhart J, Stojevic V, Green A G and Severini S 2018 arXiv:1804.03680
- [27] Melnikov A A, Nautrup H P, Krenn M, Dunjko V, Tiersch M, Zeilinger A and Briegel H J 2018 *Proc. Natl Acad. Sci.* **115** 1221–6
- [28] Huggins W, Patel P, Whaley K B and Stoudenmire E M 2018 arXiv:1803.11537
- [29] Benedetti M, Grant E, Wossnig L and Severini S 2018 arXiv:1806.00463
- [30] Bukov M, Day A G, Sels D, Weinberg P, Polkovnikov A and Mehta P 2018 *Phys. Rev. X* **8** 031086
- [31] Niu M Y, Boixo S, Smelyanskiy V and Neven H 2018 arXiv:1803.01857
- [32] Breuer H-P and Petruccione F 2002 *The Theory of Open Quantum Systems* (Oxford: Oxford University Press)
- [33] Rivas A and Huelga S F 2012 *Open Quantum Systems* (Berlin: Springer)
- [34] Ishizaki A and Fleming G R 2009 *J. Chem. Phys.* **130** 234111
- [35] Pollock F A, Rodríguez-Rosario C, Frauenheim T, Paternostro M and Modi K 2018 *Phys. Rev. A* **97** 012127
- [36] August M and Ni X 2017 *Phys. Rev. A* **95** 012335
- [37] Ostaszewski M, Miszczyk J, Sadowski P and Banchi L 2018 arXiv:1803.05193
- [38] Gorini V, Kossakowski A and Sudarshan E C G 1976 *J. Math. Phys.* **17** 821
- [39] Lindblad G 1976 *Commun. Math. Phys.* **48** 119
- [40] Ciccarello F, Palma G and Giovannetti V 2013 *Phys. Rev. A* **87** 040103
- [41] Ziman M, Štelmachovič P, Bužek V, Hillery M, Scarani V and Gisin N 2002 *Phys. Rev. A* **65** 042105
- [42] Giovannetti V and Palma G M 2012 *Phys. Rev. Lett.* **108** 040401
- [43] Rybár T, Filippov S N, Ziman M and Bužek V 2012 *J. Phys. B: At. Mol. Opt. Phys.* **45** 154006
- [44] McCloskey R and Paternostro M 2014 *Phys. Rev. A* **89** 052120
- [45] Ciccarello F 2017 *Quantum Meas. Quantum Metrol.* **4** 53
- [46] Campbell S, Ciccarello F, Palma G M and Vacchini B 2018 *Phys. Rev. A* **98** 012142
- [47] Binder F C, Thompson J and Gu M 2018 *Phys. Rev. Lett.* **120** 240502
- [48] Cerrillo J and Cao J 2014 *Phys. Rev. Lett.* **112** 110401
- [49] Pollock F A and Modi K 2018 *Quantum* **2** 76
- [50] Howard M, Twamley J, Wittmann C, Gaebel T, Jelezko F and Wrachtrup J 2006 *New J. Phys.* **8** 33
- [51] Yuen-Zhou J, Krich J J, Mohseni M and Aspuru-Guzik A 2011 *Proc. Natl Acad. Sci.* **108** 17615
- [52] Bellomo B, De Pasquale A, Gualdi G and Marzolino U 2010 *J. Phys. A: Math. Theor.* **43** 395303
- [53] Wiebe N, Granade C, Ferrie C and Cory D G 2014 *Phys. Rev. Lett.* **112** 190501
- [54] Breuer H-P, Laine E-M, Piilo J and Vacchini B 2016 *Rev. Mod. Phys.* **88** 021002
- [55] Banchi L, Costagliola G, Ishizaki A and Giorda P 2013 *J. Chem. Phys.* **138** 05B609_1
- [56] Mukamel S 1999 *Principles of Nonlinear Optical Spectroscopy* (Oxford: Oxford University Press)
- [57] Zhang W-M, Lo P-Y, Xiong H-N, Tu M W-Y and Nori F 2012 *Phys. Rev. Lett.* **109** 170402
- [58] Jang S and Silbey R J 2003 *J. Chem. Phys.* **118** 9312
- [59] Chruściński D and Kossakowski A 2010 *Phys. Rev. Lett.* **104** 070406
- [60] Schmidhuber J 2015 *Neural Netw.* **61** 85
- [61] Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H and Bengio Y 2014 arXiv:1406.1078
- [62] Hochreiter S and Schmidhuber J 1997 *Neural Comput.* **9** 1735
- [63] Chung J, Gulcehre C, Cho K and Bengio Y 2014 arXiv:1412.3555
- [64] Lipton Z C, Berkowitz J and Elkan C 2015 arXiv:1506.00019
- [65] Vacchini B, Smirne A, Laine E-M, Piilo J and Breuer H-P 2011 *New J. Phys.* **13** 093004
- [66] Schmidt M and Lipson H 2009 *Science* **324** 81

- [67] Süli E and Mayers D 2003 *An Introduction to Numerical Analysis* (Cambridge: Cambridge University Press)
- [68] Breuer H-P, Kappler B and Petruccione F 1999 *Phys. Rev. A* **59** 1633
- [69] Chruściński D and Maniscalco S 2014 *Phys. Rev. Lett.* **112** 120404
- [70] Bylicka B, Chruściński D and Maniscalco S 2014 *Sci. Rep.* **4** 5720
- [71] Iles-Smith J, Dijkstra A G, Lambert N and Nazir A 2016 *J. Chem. Phys.* **144** 044110
- [72] Cui J, Cirac J I and Banuls M C 2015 *Phys. Rev. Lett.* **114** 220601
- [73] Vidal G 2008 *Phys. Rev. Lett.* **101** 110501
- [74] Frigola R, Chen Y and Rasmussen C E 2014 *Advances in Neural Information Processing Systems* 27 (Red Hook, NY: Curran Associates) pp 3680–8
- [75] Kingma D P and Ba J 2014 arXiv:1412.6980
- [76] Krogh A and Hertz J A 1992 *Advances in Neural Information Processing Systems* 4 (Burlington, MA: Morgan-Kaufmann) pp 950–7
- [77] Chollet F *et al* 2015 Keras <https://keras.io>
- [78] Abadi M *et al* 2016 12th USENIX Symposium on Operating Systems Design and Implementation OSDI 16 (Savannah, GA: USENIX Association) pp 265–83